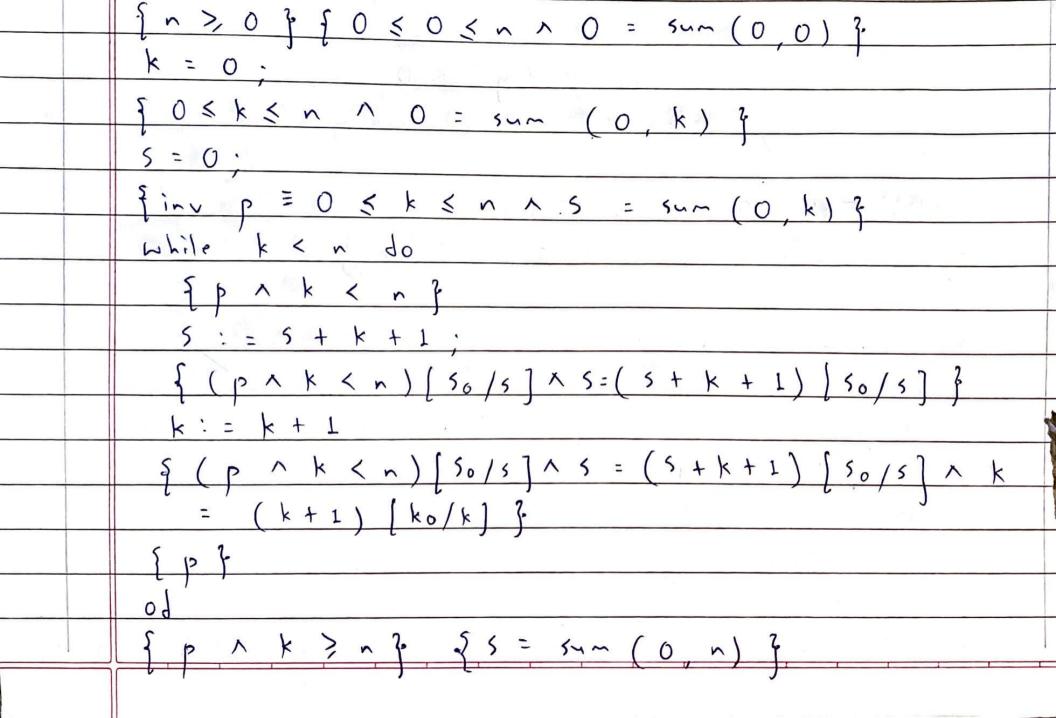
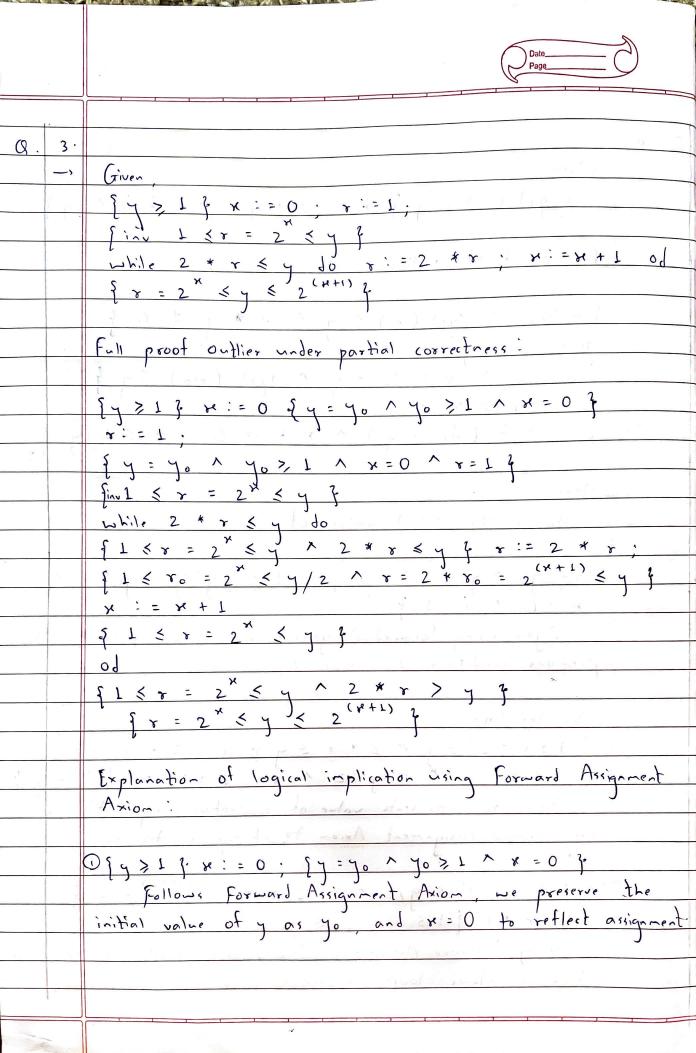
Name : Deep Pawar CWID: A 20545137 Date_____ Sub: CS536 SOP Assignment 5 1. For formal proof in Assignment 4 Question 10: al create a corresponding full proof outline under partial - full proof outline under partial correctness: Jinu $p = 1 < k < n \wedge n' = n!$ while k > 1 do $\begin{cases} p \wedge k > 1 \end{cases}$ $\begin{cases} p \left[x * k / s' \right] \left[k - 1 / k \right] \end{cases} k := k - 1$ $s' := x * k \end{cases} p \end{cases}$ od $\begin{cases} p \wedge k < 1 \end{cases} \qquad \begin{cases} q \times m = n! \end{cases}$ b) Minimal proof ontline under partial correctness: $\begin{cases} 1 & \text{int } p = 1 < k < n \land y = n \end{cases} + k \cdot k \cdot k$ while k > 1 do

k:=k-1; re:= re * k fr= n1 food a series of test pass

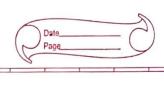
9 ~ 7 0 4 k := 0 ... = 0 ...{ in p = 0 < k < n 1 5 = sum (0, k) } while k < n do s:= s+k+1 , k:= k+1 § S = 5~~ (0, ~) } We need to calculate full proof of outline under partial correctness using backward assignment for assignments before loop & the forward assignment in the loop body:





@ [y=90 / yo > 1 / x=0 } v=1; [y=90 / yo>1 ~ Again using forward Assignment Axion, we preserve the Previous conditions & add r = 1. 3 In the loop: $\begin{cases} 1 \le x = 2^{x} \le y \land 2 * x \le y \end{cases} \quad x = 2 * x$ $\begin{cases} 1 \le x \circ' = 2^{x} \le y / 2 \land x = 2 * x \circ = 2^{(x+1)} \le y \end{cases}$ Here, so represents value of a before assignment. We use Forward Assignment Axion to change of to so. @ {1 < ro = 2 < 4/2 / r = 2 * ro = 2 (x+1) < y } x:= x + 1 { 1 < r = 2 x < y } Using Forward Assignment Axion, we update x to x+1 which maintains loop invariants.

Q. 4. -) full proof outline under total correctness for Q.3: Bound expression : 4 - 8 [y>1 } x:=0; {y>1 x x=0 } x:=1; { y > 1 ^ x = 0 ^ x = 1 } [inv 1 ≤ r = 2" ≤ y N @ bd y - r > 0 } while 2 + x < y do {1 < x = 2 x < y x 2 x x < y x y - x > 0 } {1 < 2 × ≤ y/2 ∧ x = 2 (x+1) ≤ y ∧ y-x/2 >,0 } x := x + 1 $51 \le x = 2^{x} \le y \land y - x > 0$ $\begin{cases} 1 \le x = 2^{x} \le y \land 2 + x > y \land y - x > 0 \end{cases}$ $\begin{cases} x = 2^{x} \ge y \le y \le 2 \end{cases}$ Explanation of logical implications: Q [4 > 1 } x = 0 ; [4 > 1 ^ x = 0 } Backward Assignment: (y>1) [0/k]=y>1 @ [437 VX=0] 8:=1; [437 VX=0 VX=13 Backward Assignment: (9>,1 N H=0 x T=1) [1/T] = 4>,1 N H=0



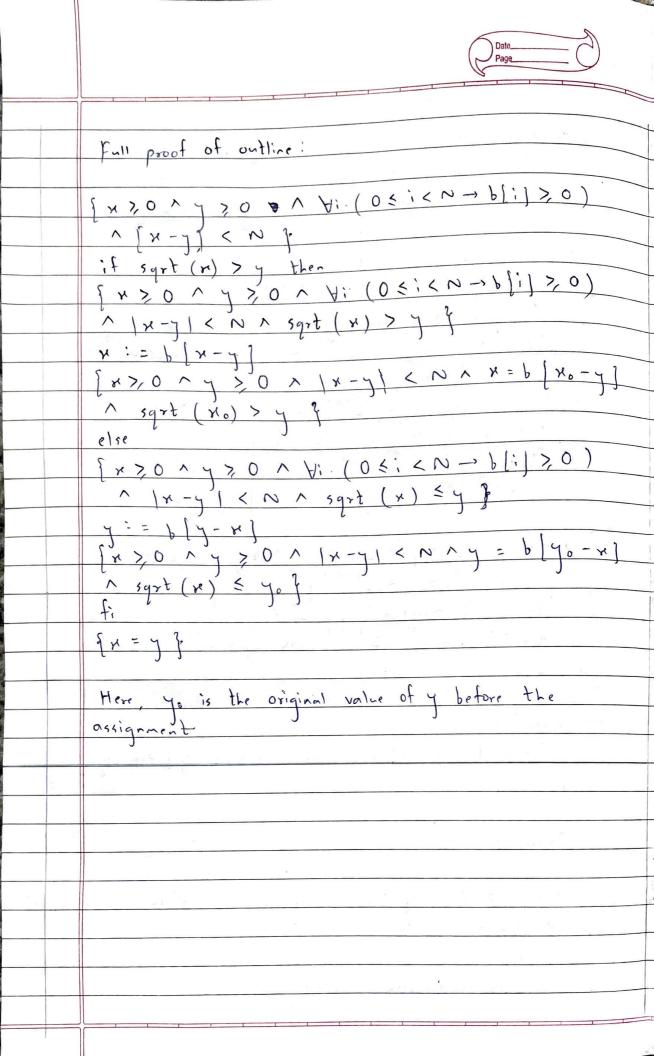
(3) {y>1 \ x = 0 \ x = 1 \} => {1 \le x = 2x \le y \ bd y - x > 0 \}

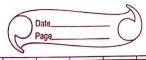
logical Implication: 1 \le x \since x = 1 \ x = 2x rsy since 1 sy (precondition) € {1 ≤ x = 2" ≤ y \ 2 * x ≤ y \ y - x > 0 } x := 2 * x .

{1 ≤ 2" ≤ y/2 \ \ x = 2 (x+1) ≤ y \ \ y - x /2 > 0 } Backward Assignment: (1<2*< y /2 x = 2 (x+1) < y x y - 8/2 > 0) [2* 8/8] (5) {1 < 2 × < y/2 × x = 2 (x+1) < y × y - x/27, 0 : 3 x: = x+1 = [15 = 2 × 5 y 1 y - 8 7 0 } Backward Assignment: (15x=2x5y /y-x>,0)[x-1/x] $0 \{1 \le x = 2^{x} \le y \land 2 * x > y \land 0 = y - x > 0 = y$ Logical Implication: $r = 2^x \le y$ (from first part) $y < 2 * r = 2^x = 2^{(x+1)}$ so $y < 2^{(x+1)}$ - Given [p] if sqrt(x) > y then x := b[x-y] else y := b [y-x] fi [x=y] Precondition P: P = x > 0 ^ 7 > 0 ^ \ (0 < i < N -> b[i] > 0) 1 x - y 1 < N

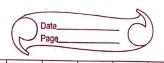
N is the size of array b

a.

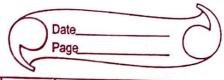




-> Given [sqrt (x) < y 3 x := x + y; x:= 1 - x [q] Full proof of outline: {x >, 0 ^ sqrt (x) < y ^ x * y # 0 } x := x * y; $\begin{cases} x > 0 \land x = x_0 * y \land iqrt(x_0) \le y \end{cases}$ $x := 1 \div x$; $\begin{cases} x = 1 \div (x_0 * y) \land x * > 0 \land x \le 1/y \end{cases}$ xo is the initial value of x.



Let of p then Id & M (W, o) Explanation. If p is an invariant, then it ensures specific constraints on the top loop's execution preventing the loop from entering states where by is violed or program does not terminate properly. Thus convergence is guaranteed by the existence of bound expression. b) The value of t can be negative after the execution of the last iteration of W. Explanation: If t is used as a bound expression for the loop W, it must remain non-negative throughout the loop's execution to guarantee proper termination. J sp (p ∧ B ∧ t = to s) => t < to Explanation: If sp (p ABA + = to s) holds meaning the program is in the same state where t = to at the start of the current iteration, then after executing the body of the loop to must be less than to due to the decreasing nature of the bound variable t. 3) p > + > 0 => B Explanation: The fact that t > 0 does not necessarily imply that B holds & thus does not guarantee another iteration. The loop could terminate due to reasons other than I reaching zero.



_	==	
	-	
	e	t < 0 => ¬p
	1 - 1	True
1.		Explanation: It is contra-positive of p => t >0 which
		guaranteed by the definition of bound expression since t
		is a bound expression it should never be negative thus
	the section of the se	+ < 0 is fabre & False implies anything is True.
۶.	8.	
	·al	x = k + n
		No 2
		Explanation: without evidence that x-k +n >, o at the
		beginning, we cannot consider re-k+n is a valid bound
	- To	franking for la

Explanation: n-k satisfies all the conditions of a bound function for the loop W, as it starts non negative, decreases with each iteration & reaches zero or negative upon loop termination.

c/n-k+C -> Yes Explanation: Since n-k + c is non-negative initially, decreases

by a fixed amount with each iteration, & will eventually

reach zero, n-k + c can be a valid bound function for w. d) -k-c -> No Explanation: Because k- (increase in each iteration rather than decreasing, it cannot be used as a bound function



e j 2° · 2 c - k

-> Yes

Explanation: since 2° 2' = 2 n+c-k is positive initially, decreases by a factor of 2 in each iteration & approaches zero as the loop progresses, it can be considered a valid bound function for W.

g. 3

Below are the 5 possible candidates for the loop invariant p & their corresponding loop condition B:

Here, we are using u as the fresh variable:

 $\bigcirc P_1 \equiv y > u \wedge x' = 2 * y < n < 3 * (y + 1), and <math>B_1 \equiv u \neq 0$

3 p2 = y > 0 A x = u * y < n < 3 * (y+1), and B2 = u ≠ 2

5 ps = y > 0 1 x = 2 * y < n < 3 * (y+u), and Bs = u = 1

-> Below are the 4 possible candidates for the loop invariant

p & their corresponding loop condition B:

 $\begin{array}{c}
\mathbb{O} \quad P_1 \equiv (\mathbf{Z} = 2^7) \wedge (2^7 \leq \aleph) \wedge (\aleph < 2^{7+1}), \text{ and} \\
\mathbb{B}_1 \equiv \gamma < 0.
\end{array}$

②
$$p_2 \equiv (y > 0) \land (2^{y} \leq \varkappa) \land (\varkappa < 2^{y+1}), \text{ and}$$
 $g_2 \equiv z \neq 2^{y}$

(3)
$$P_3 = (y > 0) \land (Z = 2^{y}) \land (x < 2^{y+1}), and$$